Remarks

All claims in prosecution 1 through 8 stand rejected under 35 USC 103 as obvious in view Lomet (US patent 6,067,550). The rejections are respectfully traversed.

Lomet teaches a recovery scheme that renders data records and application programs persistent across system crashes. Page-oriented, database style database recovery is extended to application programs.

An application program's state is treated as a single cached object, akin to a single memory page, which can be atomically flushed to a stable database. Application executions occurring between resource manger interactions are mapped to loggable operations and are posted to a stable log (see column 9, lines 7-18).

The Lomet disclosure that appears most relevant to the claimed invention · appears to be in Figures 24, 25 and 27. These Figures show portions of a stable log that is analogous to the log that is updated in the claimed invention. However, <u>where the claimed invention is concerned with creating the records in the log in a way that is safe from corruption, Lomet discloses nothing about how its log records are created. That is not the concern of the patent. Lomet is concerned with using those records to restore an application state after failure, not with how they are created in the first place.</u> Thus, it must be assumed that Lomet uses any of several known techniques to create the records. Such known techniques are summarized in the specification and will not be repeated here.

The logging operations disclosed by the Lomet can in no way be likened to that taught by the present invention.

The present invention purposely introduces an element of redundancy by using the data in the buffer to produce two <u>identical</u> pages partially full of the <u>same</u> data

Serial No. 09/898,354   2

before destroying one copy of the data. By employing redundancy it is possible to ensure that a good copy of the most recent update to the log is always retained. There is less chance that data will become corrupted during a write to the log leaving no recent copy of the data from which to recover.

Thus at the earliest opportunity two identical partial pages are established - that is two pages comprising the same data (see figure 10).

The Examiner likens this to the text at column 6, lines 20 to 25. This paragraph can not be taken to mean the same thing. This paragraph states that an application state is treated as a single object that can be atomically flushed to the stable database. This means that data (i.e. an application state) is written out to non-volatile storage at one attempt (atomically). But again, how the log is protected during this flush is not discussed.

There is no disclosure in this paragraph as to what the flushed data looks like in the log/how the log manager writes the data to the log. One would assume that the single object would consume as many pages in the log as are necessary to hold the complete object.

The paragraph goes on to say that "Application operations often cause changes to the data pages, records or other types of objects stored in volatile cache. The modified objects that result from application operations are from time to time flushed to the stable database".

But again, there is no disclosure or suggestion of establishing two identical partial pages (containing the same data). The preceding paragraph simply means that application state will change as a result of application operations and that these changes need to be reflected in non-volatile storage. Once again this extract provides no detail as to how the data is written/stored in the log.

Serial No. 09/898,354   3

·The Examiner further states that the disclosure at column 5, line 61 to column 6, line 4 is equivalent to steps two and three of the present claim 1.

Once again, there does not appear to be any reasonable comparison between the two. While this extract mentions a stable log holding log records, it does not go into any more detail. There is no disclosure as to how data is actually stored in the log, let alone that a data segment D is partitioned into two segments D1 and D2 and that a page I is filled with a first write operation of its present contents concatenated with D1.

The Examiner further asserts that the step of creating identical partial pages I+1 and I+2 with a single, second write operation of D2 to both pages is taught by column 7, lines 10-15.

It is respectfully submitted that it is not possible to derive the final step of the present claim 1 from col. 7 of Lomet..

The final step of the present invention is <u>not</u> concerned with flush order dependencies. Rather it is concerned with creating two <u>identical</u> partial pages (containing the <u>same</u> data) in positions I+1 and I+2 - i.e. introducing the same element of redundancy that the algorithm started with so that the process can begin all over again.

Regarding the Examiner's reference to column 33, line 60 to column 34, line 9 - this passage explains how different states of the application object are recorded in the log (States n; n+g; and n+h). These are final log states. Once again, there is no teaching of how they are reliably generated; nor does this teach the step of establishing identical partial pages.

The invention is entirely different from Lomet and can be best understood with reference to figure 10.

Serial No. 09/898,354   4

The present invention is concerned with how to record data to non-volatile storage using a method which saves on the number of disk writes necessary while helping to avoid the possibility of data corruption such that recovery is not possible.

By writing at "write 1" the complete page (containing C2 and D1) to position I, if any corruption occurs during the writing there should still be a good copy of C2 at position I+1. This is due to the element of redundancy that has been introduced by the step of establishing to identical partial pages (each containing data fragment C2) at the earliest opportunity.

Having written the complete page (during write 1) to position I, two new partial pages can be established (each containing data fragment D2) at position I+1 and I+2. This can be achieved via a single write (write 2) by producing two copies of the data in the buffer and writing both copies out to non-volatile storage. Thus the whole process can start all over again (i.e. two identical partial pages have been established as described by step 1 of the present claim 1).

In this way, a good copy of the most recent version of the page is maintained using the minimum number of writes (i.e. only 2 in the case of figure 10). The other (prior art) solution described with reference to figure 4 requires an extra write (write 3) in the same situation in order to maintain data integrity.
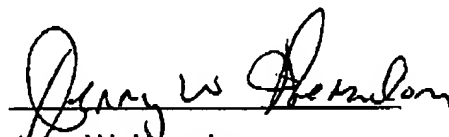
Having thus shown that there is no relation between Lomet and the claimed invention, Examiner is requested to reconsider this application and pass it to issue.

Serial No. 09/898,354   5

A prior art solution is shown by figure 4. In this solution there is only one copy of the partial page 200 containing data A (before state). When an additional piece of data B is received, it is too large to fit into the complete page and is thus split into data fragments B1 and B2. B1 is used to complete the page and the complete version of the page is written out to position N+1 (write 1). It could be written straight to back to position N, but in doing this there is a danger that the only good copy of A will be corrupted. Thus the page is written to N+1 and then copied back to position N (write 2). At write 3 a new partial page is created in position N+1.

Thus a good copy of the most recent page state is retained and the updated version written into another page.

Figure 5 shows that it would also be possible to combine writes 2 and 3 into a single write - i.e. to write the complete copy of the page 200(ii) back into position N and at the same time to write the next partial page 210 into position N+1. As explained, with reference to figure 6 this also has the danger of data corruption.

Respectfully Submitted,

Jerry W. Herndon
Reg. No. 27,901
Attorney for the Applicants

RECEIVED
CENTRAL FAX CENTER

OCT 1 4 2003

OFFICIAL

IBM Corporation, IP Law Dept. T81
3039 Cornwallis Road
PO Box 12195
Research Triangle Park, NC   27709-2195
Telephone: 919-543-3754
Fax: 919-254-4330

Serial No. 09/898,354   6